What is claimed is:

1. A system for detecting termination of an application instance using locks, comprising:

a holding child process forked from a parent process, the holding child process comprising a connection to a monitored application instance and an exclusive lock on the monitored application instance, the holding child process returning a ready signal upon successfully acquiring the exclusive lock;

a waiting child process forked from the parent process subsequent to the holding child process, the waiting child process comprising a connection to the monitored application instance, the waiting child process blocking on the exclusive lock on the monitored application instance and returning a result signal upon at least one of acquiring the exclusive lock and clearing the block on the exclusive lock; and

the parent process processing the return signal.

2. A system according to Claim 1, further comprising:

the parent process processing a standard error received from the waiting child process.

3. A system according to Claim 1, further comprising:

the parent process processing a non-standard error received from the waiting child process.

4. A system according to Claim 3, further comprising:

a validation module checking for termination of the monitored application and signaling termination of the monitored application to a cluster service.

5. A system according to Claim 3, further comprising:

a validation module checking for termination of the monitored application and restarting the holding child process and the waiting child process.

6. A system according to Claim 1, wherein the application instance comprises a database server instance.

7. A method for detecting termination of an application instance using locks, comprising:

starting a holding child process from a parent process, comprising:

connecting to a monitored application instance;

acquiring an exclusive lock on the monitored application instance; and

returning a ready signal upon successfully acquiring the exclusive lock; and

starting a waiting child process from the parent process subsequent to the holding child process, comprising:

connecting to the monitored application instance;

blocking on the exclusive lock on the monitored application instance; and

returning a result signal upon at least one of acquiring the exclusive lock and clearing the block on the exclusive lock; and

processing the return signal at the parent process.

8. A method according to Claim 7, further comprising:
processing a standard error received from the waiting child process.

9. A method according to Claim 7, further comprising:
processing a non-standard error received from the waiting child process.

10. A method according to Claim 9, further comprising:
checking for termination of the monitored application; and
signaling termination of the monitored application to a cluster service.

11. A method according to Claim 9, further comprising:
checking for termination of the monitored application; and
restarting the holding child process and the waiting child process.

12. A method according to Claim 7, wherein the application instance comprises a database server instance.

13. A computer-readable storage medium holding code for performing the method according to Claim 7.

14. A system for detecting termination of a database instance using events, comprising:

a waiting subroutine forked from a main routine, the waiting subroutine comprising a connection to a monitored database instance and blocking on a named event in the database instance and returning a result to the main routine upon an occurrence of the named event; and

the main routine processing the result.

15. A system according to Claim 14, wherein the named event comprises a membership change event, further comprising:

the main routine processing the membership change event and restarting the waiting subroutine.

16. A system according to Claim 14, further comprising:

the main routine processing a standard error received from the waiting subroutine.

17. A system according to Claim 14, further comprising:

the main routine processing a non-standard error received from the waiting subroutine.

18. A system according to Claim 17, further comprising:

a validation module checking for termination of the monitored named instance application and signaling termination of the monitored named instance application to a cluster service.

19. A system according to Claim 17, further comprising:

a validation module checking for termination of the monitored named instance application and restarting the waiting subroutine.

20. A method for detecting termination of a database instance using events, comprising:

starting a waiting subroutine from a main routine, comprising:

connecting to a monitored database instance;

blocking on a named event in the database instance; and

returning a result to the main routine upon an occurrence of the named event; and

processing the result at the main routine.

21. A method according to Claim 20, wherein the named event comprises a membership change event, further comprising:

processing the membership change event at the main routine; and

restarting the waiting subroutine.

22. A method according to Claim 20, further comprising:

processing a standard error received from the waiting subroutine.

23. A method according to Claim 20, further comprising:

processing a non-standard error received from the waiting subroutine.

24. A method according to Claim 23, further comprising:

checking for termination of the monitored named instance application; and

signaling termination of the monitored named instance application to a cluster service.

25. A method according to Claim 23, further comprising:

checking for termination of the monitored named instance application; and

restarting the waiting subroutine.

26. A computer-readable storage medium holding code for performing the method according to Claim 20.